Scheduling Survivability-Heterogeneous Sensor Networks for Critical Location Surveillance

JIE TIAN, TAN YAN, XIN GAO, and GUILING WANG, New Jersey Institute of Technology

Sensor nodes deployed outdoors for field surveillance are subject to environmental detriments. In this article, we propose a heterogeneous sensor network composed of sensor nodes with different environmental survivability to make it robust to environmental damage and keep it at a reasonable cost. We, for the first time, study the scheduling problem in such heterogeneous sensor networks for critical location surveillance applications. Our goal is to monitor all the critical points for as long as possible under different environmental conditions. We identify the underlying problem, theoretically prove its NP-complete nature, and propose a novel adaptive greedy scheduling algorithm to solve the problem. The algorithm incorporates several heuristics to schedule the activity of both regular and robust sensors to monitor all the critical points, while at the same time minimizing and balancing the network energy consumption. Simulation results show that our algorithm efficiently solves the problem and outperforms other alternatives.

Categories and Subject Descriptors: C.2.2 [**Computer-Communication Networks**]: Network Protocols

General Terms: Algorithm, Design, Performance

Additional Key Words and Phrases: Heterogeneous wireless sensor networks, scheduling algorithm, critical location surveillance

ACM Reference Format:
Jie Tian, Tan Yan, Xin Gao, and Guiling Wang. 2015. Scheduling survivability-heterogeneous sensor networks for critical location surveillance. ACM Trans. Sensor. Netw. 11, 4, Article 56 (July 2015), 23 pages.
DOI: http://dx.doi.org/10.1145/2764914

1. INTRODUCTION

Wireless sensor networks are tightly coupled with the physical world where they are deployed for different applications, such as field surveillance. As a tiny electronic device that usually operates outdoors, an on-duty sensor node is vulnerable to many environmental attributes or detriments, such as rain or snow [Ruiz et al. 2004]. The problems have been frequently cited as a vital reason for sensor nodes' being unreliable, but very few research efforts have been dedicated to addressing the problem directly or fundamentally.

To increase the survivability of a sensor node outdoors, one way is to equip it with additional protections to make it, for example, waterproof [Martinez et al. 2004]. However, this may greatly increase the cost of sensor networks, making them less applicable [Selavo et al. 2007]. Also, many environmental conditions that can do harm to sensor nodes only happen occasionally in an area. For example, when the probability of rain is not high, to make all sensor nodes waterproof is not cost-effective. Hence, this article

This work is supported by the National Science Foundation grant NSF-1128369.
Authors' address: J. Tian, T. Yan, X. Gao, and G.Wang, Computer Science Department, New Jersey Institute of Technology, GITC Building, Room 4400, University Heights, Newark, NJ 07102-1982, USA; emails: {jt66, ty7, xg54, gwang}@njit.edu.

DOI: http://dx.doi.org/10.1145/2764914

proposes to construct a heterogeneous sensor network composed of both regular sensor nodes lacking of protections and sensor nodes with additional environmentally robust features, such as waterproofing [Tian et al. 2014b]. We call the latter environmentally robust nodes, or robust nodes for short. The term environmental survivability characterizes such heterogeneity, which is fundamentally different from the widely studied heterogeneity in terms of computational capacity, energy, communication range, and sensing range [Cardei et al. 2008; Du and Lin 2005; Wang et al. 2008, 2011]. To study this type of problem, this article analyzes the most common environmental attribute, rain, as an example, and robust sensors are assumed to be waterproof. The research results of the article can be easily extended to other environmental survivability.

This article focuses on employing such kind of heterogeneous sensor networks for outdoor field surveillance, an important application in both civilian and military sectors. In particular, we target the critical points surveillance application [Cardei et al. 2005; Liu et al. 2005]. That is, in a target field, there are critical locations that need constant monitoring by some sensors even though it is raining. Our goal is to schedule the activity of both regular and robust sensors to monitor all the critical points as long as possible under different weather conditions. Note that regular sensors cannot work in the rain; otherwise they will be damaged permanently and cannot work anymore though they still have energy left. Robust sensors should be conservative in energy consumption such that they still have enough energy left and can work long enough when there comes rain. On the other hand, regular sensors should not be overly used considering rain may not come for a long time and overusing one portion of the sensors can result in energy imbalance and shorter network lifetime. Because of this additional dimension of complexity in sensor scheduling incurred by weather, previous works [Cardei et al. 2005; Liu et al. 2005; Shih et al. 2009] in studying activity scheduling for trade-offs between sensor coverage and network lifetime cannot be employed here. To the best of our knowledge, this article for the first time studies the scheduling problem in heterogeneous sensor networks regarding environmental survivability for critical location surveillance applications.

We leverage the widely available weather forecast information, which provides the probabilities of precipitation in coming time slots. Previous work [Kang et al. 2008] shows that even by simply adopting an inaccurate weather forecast in the scheduling, the network performance can be improved significantly. Based on the weather forecast, we introduce the Poisson model of sensor nodes' survivability under different weather conditions. We then define a new metric called *Critical Point Coverage Probability* (CPCP) to quantify the surveillance quality of a sensor network, which is the probability that all the critical points are covered at the end of each time slot. More specifically, assuming at the beginning of each time slot, all critical points are covered by some sensors, CPCP measures the probability that the points are still covered by the same or less number of sensors at the end of the time slot considering some sensors may die. We want to maintain CPCP to a required level, and we define the network lifetime as the duration that CPCP is kept at that level. The aim of this article is to schedule sensors in each time slot based on the weather forecast information, such that the network lifetime is prolonged as much as possible. To increase the network lifetime, in each time slot, under the required CPCP, we minimize the number of selected on-duty sensors while maximize the remaining energy of the scheduled sensors.

To tackle the problem, we first employ weighted bipartite graph as the data structure, and formulate the problem as a *β-Coverage Problem*. We theoretically prove that the β-Coverage Problem is NP-complete since the Set Cover Problem [West 2001], a well-known NP-C problem, can be reduced to it. We then propose a novel adaptive greedy scheduling algorithm to tackle our NP-C problem. Given the weather forecast, the algorithm first filters out all the sensors that are likely to fail in the coming slot. Then, among all the remaining sensors, the algorithm always selects the one that covers the greatest number of critical points. This procedure repeats until all the critical points are covered by the selected sensors. If the selected sensors cannot provide a required CPCP, several heuristics are employed to add as few as possible extra sensors to the selection to increase the CPCP. For example, we check the coverage of all the critical points, and always select the sensor that can cover the greatest number of critical points, which need more sensors to cover. To balance the energy consumption, we avoid overusing a single type of sensor. The algorithm will be launched multiple times to calculate multiple sets of sensors that can provide a required CPCP. Among all the selected sets, we only power on the one with the highest total remaining energy. In this way, the overall network energy consumption is balanced. We execute the algorithm periodically with a new incoming weather forecast. Simulation results show that our adapted greedy heuristic solves the problem efficiently, and outperforms other alternatives.

To summarize, our contribution in this article is threefold:

—We, for the first time, study the scheduling algorithm in a heterogeneous wireless sensor network with sensors having different capabilities in withstanding a harsh environment.
—We identify the underlying scheduling problem and theoretically prove it is NP-complete.
—An adaptive greedy heuristic is proposed to obtain an efficient and effective solution to the problem.

The remainder of the article is organized as follows. We present the assumptions and models in Section 2. We formulate the problem and prove its NP-completeness in Section 3. Section 4 describes the proposed adapted greedy heuristic and the algorithm analysis is presented in Section 5. Section 6 reports simulation results. Related work is discussed in Section 7. Finally, we conclude the article in Section 8.

2. ASSUMPTION AND MODEL

2.1. Weather Model

Current weather forecast techniques are mature and can provide accurate weather forecast information of a given area [Lorenc 1986; Richardson 2007]. In the article, we assume a weather forecast is available. Specifically, a weather forecast provides the probabilities of precipitation in the next multiple time slots. Considering the weather forecast is not always correct and the historical forecast accuracy is available, we also assume the associated accuracy of the weather forecast estimated from the historical data is available.

Consider a weather forecast with a precipitation chance f and its forecast accuracy u. If $f \geq 30\%$, "rainy" will be reported and "no rain" otherwise [Enyedi 2010]. The forecast has a probability $1-u$ to report no rain when the actual weather is rainy, and reports rainy when there is no rain, which are Type I error and Type II error according to statistical test theory [Peck and Devore 2013]. Incorporating both the forecast and the potential errors, we use $fu + (1-f)(1-u) \geq 30\%$ as the condition to determine an incoming rain in our algorithm. Note that, although a weather forecast serves as an important parameter in our algorithm, our algorithm does not rely on a very high accuracy of weather prediction, which is shown in the performance evaluation section of this article.

2.2. Sensor Model

We assume every sensor knows its location, which can be obtained by GPS or localization methods [Chen et al. 2011b; Huang et al. 2010; Liu et al. 2012]. There are two types of sensors in the network: regular sensors and robust sensors. A robust sensor is reliable in both sunny and rainy weather. A regular sensor can only work well when there is no rain and has a high probability to fail in the rain. If a sensor fails, it cannot function again. An isotropic sensing model is adopted [Hu et al. 2014; Chen et al. 2011a]. The sensing area of each sensor node is a circle with the same radius. If the occurrence of the event is within the sensing range of a sensor node, then the event is assumed to be detected, otherwise not. It is also well known as 0/1 model.[1]

We adopt the widely accepted Poisson distribution [Hoblos et al. 2000] to model the survivability of a sensor. The survival probability[2] of a sensor at time t is the probability that it does not fail during time interval $[0, t]$, as shown next:

$$w(t) = e^{-\lambda t}, \tag{1}$$

where λ is the failure rate.

Sensor's failure rate is a constant value provided by its manufactory [Aschenbrenner 2006; Goble 2002]. We assume such value can be obtained directly from sensor's manual. Since the survivability of a robust sensor in sunny weather and in rainy weather is the same, a constant $\mathbf{C}$ denotes its failure rate λ. For a regular sensor, in sunny weather its survivability is the same as that of a robust sensor, while in rainy weather, a different constant $\mathbf{C}'$ represents its failure rate[3]. Consider a weather forecast with precipitation chance f and associated forecast accuracy u, based on the previous discussion, a regular sensor's failure rate is set in our algorithm as follows:

$$\lambda = \begin{cases} \mathbf{C} & \text{if } fu + (1-f)(1-u) < 30\%, \text{ no rain} \\ \mathbf{C}' & \text{otherwise, rainy.} \end{cases} \tag{2}$$

When there are other environmental detriments, such as frozen and forest fire, we only need to redefine the sensor failure rate and corresponding survival model for each detriment. Then, the models under such detriment can be incorporated in the following algorithm.

2.3. Surveillance Model

We concentrate on critical location surveillance in the article. In the monitored field, there are known *critical points* where events of interest are expected to occur. To capture all the events, regular sensors and robust sensors are deployed in the network to cover all the critical points. For example, in Figure 1, regular sensor s_1 covers critical points r_1 and r_2, robust sensor s_2 covers r_1 and r_3, and regular sensor s_3 covers r_2 and r_3.

[1] For other models like probability sensing models, they still work effectively. One possible solution is to set a threshold as the positive detection probability in the probability models, which can be used to determine whether or not a critical point is covered.

[2] Throughout the article, we use "survivability" and "survival probability" interchangeably.

[3] When a sensor encounters a rainy period and a sunny period, the calculation procedure is as follows: the failure probability under rainy/sunny weather based on a failure rate of $\mathbf{C}'/\mathbf{C}$ is calculated. Two failure probabilities are added to get the cumulative failure probability after the two periods. Then the failure rate during these two periods is calculated based on the cumulative failure probability.

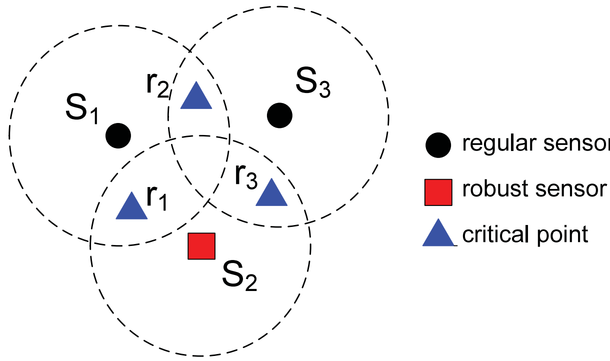


Fig. 1. Example of sensor surveillance.

Since the deployment has redundancy, to cover all the critical points, it is unnecessary to power on all the sensors at any time. Take Figure 1 as an example. Assume s_2 and s_3 are powered on during a time slot, and their survivability is w_2 and w_3 calculated by Eq. (1). Since the critical point r_3 is covered by both s_2 and s_3, the probability that r_3 can be covered by at least one sensor during the time slot is $1 - (1 - w_2)(1 - w_3)$. Similarly, the probabilities for r_1 and r_2 to be covered are w_2 and w_3, respectively.

2.4. Our Goal

Given a surveillance area and its weather forecast, our goal is to select a set of sensors to power on based on the survivability of each sensor, such that the following requirements are satisfied: (1) At the beginning of a time slot, the set of sensors should cover all the critical points. (2) The CPCP, the probability that all the critical points are still covered at the end of the time slot, should be maximized. (3) To reduce the energy consumption, the number of sensors selected to be on duty should be minimized.

Note that, it is impossible to have both maximized CPCP (requirement (2)) and the minimized number of selected sensors (requirement (3)) at the same time. Thus, we alter the requirement (2) to be *the CPCP should be higher than a threshold*. The threshold is specified by the application. Different applications may have different required CPCP. Also, the threshold can be adjusted to balance the CPCP and network lifetime.

3. THE β-COVERAGE PROBLEM

In this section, we formally define the CPCP in Definition 3.1. Based on the definition, we formulate the β-Coverage Problem, and formally prove its NP-completeness.

3.1. Problem Definition

Definition 3.1 [*CPCP*]. Consider a set of critical points R, a set of on-duty sensors S_{on}, and a set W associated with S_{on}, where $w_i \in W$ is the probability that sensor $s_i \in S_{on}$ can survive until the end of the time slot. For each critical point $r_k \in R$, we have a subset of S_{on} $(s_1, s_2, \ldots s_p)$, which are all the on-duty sensors that cover r_k at this time slot. Accordingly, we have $w_1, w_2, \ldots, w_p$ associated with $s_1, s_2, \ldots, s_p$. The coverage probability of r_k is defined as $P^r_k = 1-(1-w_1)(1-w_2)\ldots(1-w_p)$. For all the m critical points in R, the CPCP provided by S_{on} is defined as $P^r = P^r_1 \cdot P^r_2 \cdots P^r_m$.

Definition 3.2 [*β-Coverage Problem*]. Given a weighted bipartite graph $G = (V = (S + R), E)$, which contains two set of vertices, S and R, S $(= \{s_1, s_2, \ldots, s_n\})$ is the set of all the deployed sensors, and R $(= \{r_1, r_2, \ldots, r_m\})$ is set of all the critical points in the area. An edge $(s_i, r_k) \in E$ ($s_i \in S$ and $r_k \in R$) exists if and only if critical point r_k is covered by sensor s_i. W is a set of weights associated to each edge $(s_i, r_k) \in E$, which is the probability that r_k can be covered by s_i (survivability of s_i) at the end of a time unit.

Given a predefined threshold β ($0 < \beta \leq 1$), our problem is to select a set of minimum number of sensors $S' \subseteq S$, which is subjected to

—**Subject (1):** $\forall r_k \in R$, $\exists s_i \in S'$ such that $(s_i, r_k) \in E$.
—**Subject (2):** The CPCP of R provided by $S' \geq \beta$.

3.2. NPC Proof

THEOREM 3.3. *The β-Coverage Problem is NP-complete.*

PROOF OF THEOREM 3.3. To facilitate the proof, we formulate the decision version of the β-Coverage Problem as, given an integer $k \leq |S|$, is there a set of vertices $S' \subseteq S$ that solves the problem with $|S'| \leq k$?

Prove to be in NP. Given a solution S' of the β-Coverage Problem, which is a subset of vertices from S, to check the correctness of the solution, we need to determine whether that subset satisfies both Subject (1) and Subject (2) and has a size less than or equal to k. It takes $O(|S'| * |R|)$ to verify Subject (1), $O(|S'| * |R|)$ to verify Subject (2), and $O(1)$ to verify if $|S'| \leq k$. In total, it takes $O(|S'| * |R|)$ to verify the solution, which is in polynomial complexity, and thus β-Coverage Problem $\in$ NP.

Restriction. We reduce Set Cover Problem [West 2001] to β-Coverage Problem by applying *restrictions* to the problem. Given an original β-Coverage Problem, we restrict all the values in set W to be 1, which means, at the beginning of a time slot, if a sensor is selected to be on duty, it has probability 1 to survive till the end of the time unit. Accordingly, we let $\beta = 1$ in Subject (2), which means we require all the critical points to be covered at probability 1. In this case, Subject (2) becomes equivalent to Subject (1), because Subject (2) is satisfied if and only if a set of sensors are selected, such that the selected sensors cover all the critical points. Therefore, in the restricted β-Coverage Problem, we only consider Subject (1).

Reduction. Let a set B of m elements and a collection $A (= \{A_1, A_2, \ldots, A_n\})$ of n subsets of B be an arbitrary instance of the Set Cover Problem. We construct the restricted β-Coverage Problem $G = (S + R, E)$ as defined in Definition 3.2 with all the elements in W equal to 1, as follows:

—$R = B$,
—s_i has edges with $\{r_j, r_k, \ldots, r_p\}$ in G, if $A_i = \{r_j, r_k, \ldots, r_p\}$.

It is easy to see such construction can be done in polynomial time as it only requires simple component replacements. We now claim the constructed restricted β-Coverage Problem has a solution with $S' \subseteq S$ and $|S'| \leq k$, if and only if the Set Cover Problem has a solution $A' \subseteq A$ and $|A'| \leq k$.

If there is a solution $S' \subseteq S$ to the constructed restricted β-Coverage Problem with $\forall r_j \in R$, $\exists s_i \in S'$ such that $(s_i, r_j) \in E$ and $|S'| \leq k$, we construct $A' \subseteq A$ to be a collection of $|S'|$ sets as follows: for every $s_i \in S$, pick all the vertices in R that s_i connects to, and add the set of the picked vertices to A'. Since $B = R$, we have $\forall b_j \in B$, $\exists A_i \in A'$, such that $b_j \in A_j$ and $|A'| \leq k$.

Conversely, if there is a solution $A' \subseteq A$ to the Set Cover Problem with $\forall b_j \in B$, $\exists A_i \in A'$, such that $b_j \in A_j$ and $|A'| \leq k$, we construct $S' \subseteq S$ to be a set of $|A'|$ vertices as follows: for every $A_i \subseteq A'$, pick a vertex $s_i \in S$, which connects all the vertices in A_i in graph G, and add s_i to S'. Since $B = R$, we have $\forall r_j \in R$, $\exists s_i \in S'$, such that $(s_i, r_j) \in E$ and $|S'| \leq k$.

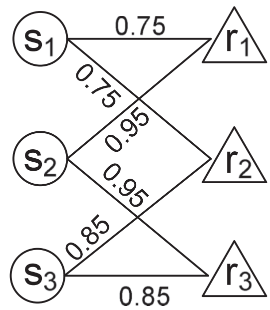


Fig. 2. An example of bipartite graph.

Conclusion. Therefore, the Set Cover Problem is reducible to the restricted β-Coverage Problem. Since all the restrictions and reductions are done in polynomial time, the original β-Coverage Problem is NP-complete. □

Take Figure 1 as an example. The surveillance scenario can be formatted using a bipartite graph as shown in Figure 2, where the survival probability of each sensor is attached to the corresponding edge. If given $\beta = 0.8$ and $k = 2$, then the set $\{s_2, s_3\}$ is a "yes" instance to the problem, which yields

$$\begin{aligned} P^r &= P_1^r \cdot P_2^r \cdot P_3^r \\ &= w_2 \cdot w_3 \cdot [1 - (1 - w_2)(1 - w_3)] \\ &= 0.95 \times 0.85 \times [1 - (1 - 0.95)(1 - 0.85)] \\ &= 0.801. \end{aligned} \tag{3}$$

3.3. Analysis of Potential Solutions

According to Theorem 3.3, the β-Coverage Problem has no polynomial solution unless $P = NP$. We show that the Set Cover Problem can be reduced to the β-Coverage Problem. However, Linear Programming [Cardei et al. 2005; Liu et al. 2005; Shih et al. 2009], the most efficient and widely applied heuristics for solving the Set Cover Problem, cannot be applied to solve the β-Coverage Problem. The reason is that in the β-Coverage Problem, to compute the CPCP, we need to multiply the coverage probabilities of all the critical points, and the coverage probability calculation for each individual critical point further involves multiplication, addition, and subtraction. To explain in detail, according to Definition 3.2, our goal is to minimize the number of selected sensors subject to two conditions. The proposed problem can be formally written in the following format:

$$\begin{aligned} \text{Minimize: } & \textstyle\sum_{i=1}^{n} x_i, \\ \text{Subject to: } & \textstyle\sum_{i \in c_k} x_i \geq 1, \forall r_k \in R, \\ & x_i \in \{0, 1\}, \\ & P^r \geq \beta, \\ & P^r = \textstyle\prod_{k=1}^{m} \left(1 - \prod_{i=1}^{n} \left(1 - x_i w_i^{(k)}\right)\right), \end{aligned}$$

where c_k is the set of sensors that can cover the critical point r_k and $w_i^{(k)}$ is the weight of sensor i that can cover the critical point r_k. $x_i = 1$ if and only if sensor s_i is in the selected set. Otherwise, $x_i = 0$. To solve the problem using linear programming, P^r must be written in the linear format. By taking the logarithm of both sides, we have $\log(P^r) = \log(P_1^r) + \log(P_2^r) + \cdots + \log(P_m^r)$. For each individual $\log(P_k^r)$ $(k = 1, 2, \ldots, m)$, it must also be in the linear format. According to the formula of P_k^r, we have $1 - P_k^r = (1 - x_1 w_1^{(k)})(1 - x_2 w_2^{(k)}) \cdots (1 - x_n w_n^{(k)})$. By taking the logarithm of both sides, we have $\log(1 - P_k^r) = \log(1 - x_1 w_1^{(k)}) + \log(1 - x_2 w_2^{(k)}) + \cdots + \log(1 - x_n w_n^{(k)})$. It is obvious that $\log(1 - P_k^r)$ cannot be linearly transformed to $\log(P_k^r)$. Since the calculation embeds nonlinear operations, it *cannot be resolved by Linear Programming*. In the next section, we propose an adaptive greedy scheduling algorithm to solve the problem.

4. ADAPTIVE GREEDY SCHEDULING ALGORITHM

This section presents an adaptive greedy scheduling algorithm to solve the β-Coverage Problem. We divide the time span into slots, and assume the weather forecast for the next several time slots can be obtained beforehand. Based on the weather forecast, we execute the algorithm to compute a schedule for each time slot, based on which, sensors are scheduled to be on duty or off duty.

The algorithm has three phases: (1) Cover Selection, (2) CPCP Enhancement, and (3) Energy Balance. We go through the first two phases iteratively to select multiple candidate sets of sensors that can provide the required CPCP and then in the third phase, pick one of them with the highest remaining energy as the output for one coming slot. More specifically, in Cover Selection phase, we first filter out the sensors that are likely to fail because of the weather. Then, among all the remaining sensors, we employ a greedy algorithm to select sensors to cover all the critical points. If the selected sensors together cannot provide the required CPCP, we move to the CPCP Enhancement phase and select extra sensors to improve the coverage probability for the critical points with weak coverage. We keep adding sensors until the required CPCP is reached. After that, a candidate set of sensors are selected. We then repeat the first two phases to select multiple candidate sets for one coming time slot. In Energy Balance phase, to prolong the network lifetime, among all the candidate sets, we pick the one with the best energy profile and schedule it in the next time slot. The details of the algorithm are presented in the following sections.

4.1. Cover Selection

For a time slot, based on the weather forecast with the precipitation probability and the associated forecast accuracy, we first determine whether it is going to rain in the slot. If yes, we filter out all the regular sensors and only choose robust sensors as *candidate sensors*; otherwise, every sensor is a candidate sensor.

We apply a greedy heuristic on candidate sensors to select a candidate set to cover all the critical points. The rule is that the sensor that can cover most critical points is always selected first. Then the sensor is deleted from the candidate sensors, and its covered critical points are deleted from the group of critical points. The process repeats until all the critical points are covered, and a candidate set of sensors that can cover all the critical points are obtained.

We check whether the candidate set can provide the required CPCP. If not, we move to the CPCP Enhancement phase to select more sensors from the candidate sensors to increase CPCP to the required level. The procedure repeats to generate more candidate sets.

Note that, in a redundantly deployed sensor network, it is highly possible that different sensors can cover the same number of critical points, which brings natural randomness to the result of the greedy selection in this phase. To be more specific, it is likely that multiple sensors cover the greatest number of critical points and the number they cover is the same. Therefore, different sensors can be selected according to the same greedy heuristic. If this phase is repeated and a different sensor is selected each time, different candidate sets of sensors may be calculated. For example, in our experiments with 100 sensors and 10 critical points, when the selection is repeated five times, five different sets are generated in most of the experiments.

4.2. CPCP Enhancement

In this phase, extra sensors are selected to meet the CPCP requirement. The selection targets on both balancing energy consumption and minimizing the number of selected sensors.

This phase has two steps: (1) selecting group and (2) selecting sensors. The first step aims at balancing the energy consumption between regular sensors and robust sensors, and determines which group of sensors should be considered as candidate sensors. It considers both historical and future weather condition to avoid overusing only one type of the sensors for a long time. Once a group is chosen, the second step will run iteratively until the required CPCP is reached. In each iteration, only one extra sensor that can improve the CPCP the most is selected.

Step 1: **Selecting Group.** Intuitively, regular sensors should always be selected when there is no rain so that robust sensors can be saved for rainy days. However, overusing regular sensors may result in unbalanced energy consumption and shorter network lifetime when there is no rain for a long time. Therefore, we introduce a dynamic priority to balance the scheduling of regular sensors and robust sensors in different weather conditions. The candidate sensors can only be the sensors from the group with higher priority.

The priority value of robust sensors is always 1, which is used as a baseline. The following function is used to model the priority of regular sensors.

$$\begin{cases} \frac{c}{x} \cdot a & \text{if } x < c \\ a & \text{otherwise,} \end{cases} \tag{4}$$

where $a = 0$ if the coming time slot is rainy, while $a = 1$ otherwise. In this way, regular sensors will never be scheduled in a time slot with rain. c is a positive constant, for example, 20. x is a positive variable and initially set to be 1. x varies with the cumulative weather conditions in the past time slots. If there is no rain in the past time slot, x increases by 1 and the priority of regular sensors gets smaller; otherwise, x reduces to half of its current value and a regular sensor's priority doubles. This means, if the weather has been sunny for more than c consecutive time slots, x becomes larger than c and the priority of regular sensors becomes the same as robust sensors. Then both regular sensors and robust sensors can be scheduled to be on duty. Otherwise, it is always regular sensors that will be scheduled to be on duty when there is no rain in the coming time slot.

The detailed priority adjustment algorithm for regular sensors is presented in Algorithm 1.

ALGORITHM 1: Priority Adjustment for Regular Sensor Group

if $fu + (1 - f)(1 - u) \geq 0.3$ **then**
  Set x to be the half of its value;
  Set $a = 0$;
else
  Increase x by 1;
  Set $a = 1$;
end if
if $x \geq c$ **then**
  Set $x = c$;
end if

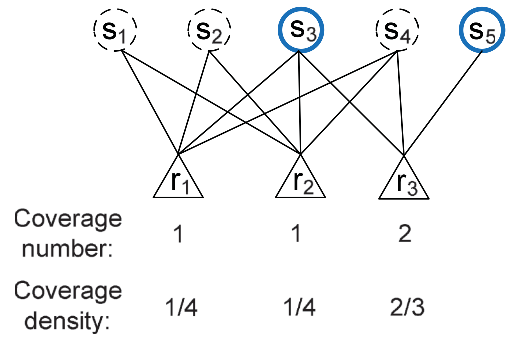


Fig. 3. Example of minimizing number of extra sensors.

***Step 2:* Selecting Sensors.** In this step, one sensor from the candidate sensors will be selected to improve the CPCP. We introduce two heuristics for the selection: coverage number based selection and coverage density based selection. The coverage number based heuristic will be applied first. If multiple sensors are selected, then the second heuristic will be applied to select only one from the multiple sensors.

Heuristic 1: Coverage Number Based Selection. For a critical point, we use *coverage number* to represent the number of sensors that are selected in Cover Selection phase to cover the critical point. We calculate the *coverage number* for all the critical points in the network. Then, from the candidate sensors picked in *Step 1*, we select the sensor that can cover most of the critical points whose coverage numbers are below the average. If only one sensor is selected, the CPCP Enhancement phase ends; otherwise, the selection is refined by applying the next heuristic.

Figure 3 shows an example with five sensors and three critical points. Suppose s_3 and s_5 are selected in Cover Selection phase to cover all the critical points. Since r_1 is only covered by s_3, its coverage number is 1. Similarly, we can calculate the coverage numbers for r_2 and r_3 are 1 and 2, respectively. r_1 and r_2 are the critical points whose coverage numbers are below average. Since all of s_1, s_2, s_4 can cover both r_1 and r_2, they all can be selected.

Heuristic 2: Coverage Density Based Selection. We define *coverage density* of a critical point as the ratio of its coverage number to the number of sensors that can cover it. In this heuristic, among the group of sensors selected by *Heuristic 1*, we select the sensor with the average coverage density of critical points that this sensor covers is the lowest and the number of critical points that it covers is the highest. For example, in Figure 3, the coverage density of r_1 is 1/4 since it can be covered by s_1, s_2, s_3, and s_4, and only s_3 is selected. Similarly, the coverage density of r_2 and r_3 is 1/4 and 2/3, respectively. $\{s_1, s_2, s_4\}$ are selected in *Heuristic 1*. Now we have s_1 covers $\{r_1, r_2\}$ and its average coverage density $(1/4 + 1/4)/2 = 1/4$. s_2 covers $\{r_1, r_2\}$ and its average coverage density $(1/4 + 1/4)/2 = 1/4$. s_4 covers $\{r_1, r_2, r_3\}$ and its average coverage density $(1/4+1/4+2/3)/3 = 7/18$. Since both s_1 and s_2 have the least average coverage density, we randomly pick one from them, and add it to the candidate set selected in Cover Selection phase.

We keep adding sensors by going through *Step 2* until the required CPCP is reached. After the CPCP Enhancement phase, a candidate set of sensors that can provide the required CPCP are selected.

4.3. Energy Balance

Multiple candidate sets are selected after we go through Cover Selection phase and CPCP Enhancement Phase iteratively. In this last phase, we select one set from the multiple candidate sets, such that the energy consumption is evenly distributed among all the sensors in the long run and no single portion of sensors are overused.

To balance the energy consumption, among all the candidate sets generated in Cover Selection phase and CPCP Enhancement phase, we always choose the set of sensors whose energy profile is the best. Specifically, for each candidate set selected in the previous two phases, we find the sensor that has the least amount of remaining energy among all the sensors in the candidate set, and mark its remaining energy, e_{min}, as the minimum remaining energy for this set. We pick the set that has the highest e_{min} to be on duty in the next time slot.

Algorithm 2 shows the whole adaptive greedy scheduling algorithm.

ALGORITHM 2: Adaptive Greedy Scheduling Algorithm

Input:
  Required CPCP: β.
  Weather forecast information for next τ periods.
Output:
  Set^{j}_{out}: the set of sensors to be scheduled at jth time slot.

for $j = 1 \to \tau$ **do**
  // Generate k candidate sets
  for $i = 1 \to k$ **do**
    // In Cover Selection phase
    Filter out sensors and generate candidate sensors Set_{CAP}.
    From Set_{CAP}, apply greedy heuristic to select a set Set_i.
    if the CPCP provided by $Set_i \geq \beta$ **then**
      record Set_i.
      continue.
    else
      // In Energy Enhancement phase
      Assign selection priority.
      Select the group of sensors with higher priority as candidate sensors Set_{CAP}.
      while CPCP of $Set_i < \beta$ **do**
        Select sensors from Set_{CAP} by Heuristic 1.
        if More than one sensor are selected **then**
          Select one of them by Heuristic 2.
        end if
        Add the sensor to Set_i.
      end while
      record Set_i.
    end if
  end for
  // In Energy Balance phase
  for $i = 1 \to k$ **do**
    Record e^{i}_{min}.
  end for
  Find $e^{h}_{min} = \max(e^{1}_{min}, \ldots, e^{k}_{min})$.
  $Set^{j}_{out} = Set_h$.
end for

5. ALGORITHM ANALYSIS

In this section, we analyze the upper bound of the number of sensors selected to achieve the required CPCP and the complexity of the proposed adaptive greedy algorithm.

5.1. Upper Bound of Number of Sensors Selected

THEOREM 5.1. *The number of sensors selected in the proposed adaptive greedy algorithm is upper bounded by*

$$\boldsymbol{R} = m\frac{ln(1-\beta^{\frac{1}{m}})}{ln(1-w_{min})}, \tag{5}$$

where w_{min} is the smallest survivability of sensors.

PROOF OF THEOREM 5.1. According to Definition 3.1, as w_i increases for any sensor i, P^r increases. As the number of critical points covered by a single sensor increases, P^r also increases. Then the worst case in the selection is the sensors with lowest survivability are selected to cover critical points and each selected sensor only covers one critical point. In this situation, the number of sensors selected is maximized.

Let w_{min} denote the lowest survivability of sensors, and d_k denote the number of sensors required to cover critical point r_k. For each critical point r_k in the worst case, we can obtain

$$P_k^r = 1-(1-w_{min})^{d_k}. \tag{6}$$

Moreover, according to the Heuristic 1 in CPCP Enhancement phase, the number of sensors covering critical points is balanced. Hence, we can obtain $d_1 = d_2 = \cdots = d_m$. Let $d = d_k$ for any k. d is the number of sensors required to cover one critical point. To reach required CPCP, the following equation can be obtained:

$$\beta = \prod_{k=1}^{m} P_k^r = (1-(1-w_{min})^d)^m. \tag{7}$$

Solving the equation, we can have

$$d = \frac{\ln(1-\beta^{\frac{1}{m}})}{\ln(1-w_{min})}. \tag{8}$$

Then, the total number of the sensors selected is

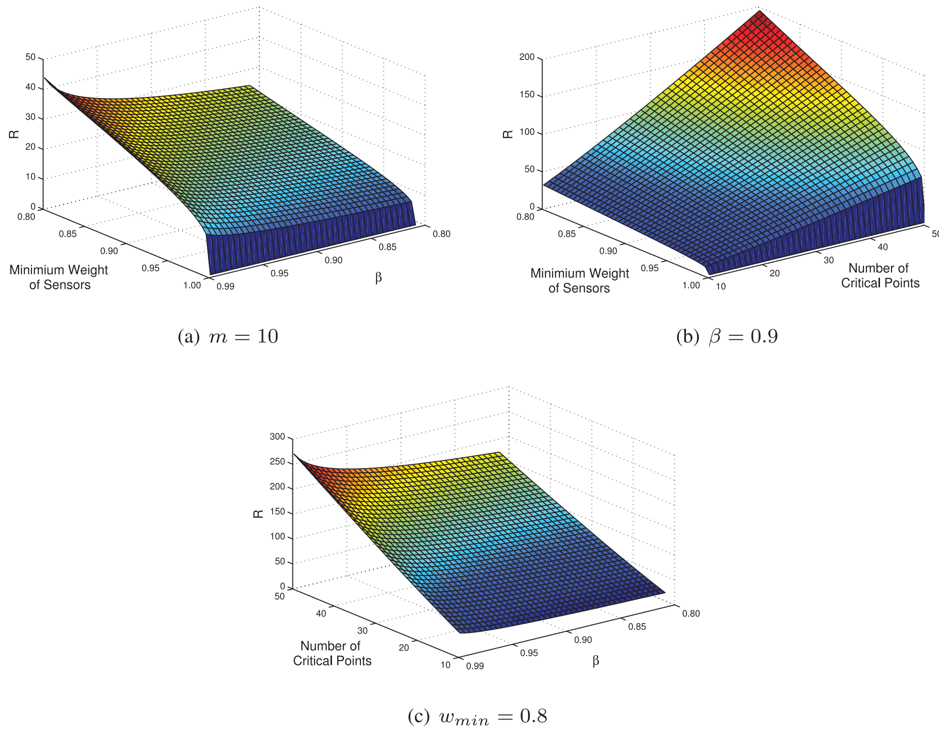
$$\mathbf{R} = md = \frac{m\ln(1-\beta^{\frac{1}{m}})}{\ln(1-w_{min})}. \tag{9}$$

Since the survivability of each sensor is at least w_{min}, the total number of sensors is upper bounded by $\mathbf{R}$ and the theorem is proved. □

The numerical results of $\mathbf{R}$ are shown in Figure 4. Because the value of $\mathbf{R}$ is determined by three parameters, m, β and w_{min}, we set one of them to be constant and vary the others in the comparisons. Figure 4(a) shows the results when $m = 10$, Figure 4(b) shows the results when $\beta = 0.9$, and Figure 4(c) shows the results when $w_{min} = 0.8$. It is shown that $\mathbf{R}$ is largely affected by w_{min} and m. Moreover, a preliminary evaluation is performed to observe the number of sensors selected in each selection and compare the algorithm's results with analytical results. Figure 5 shows the results by running the proposed algorithm for 150 times. It can be observed that the number of selections under the proposed algorithm does not exceed the theoretical results, which also verifies Theorem 5.1.

5.2. Time Complexity

The time complexity of our algorithm largely depends on the time complexity of the heuristics in Cover Selection phase and CPCP Enhancement phase (Algorithm 2).



(a) $m = 10$

(b) $\beta = 0.9$

(c) $w_{min} = 0.8$

Fig. 4. The values of $\mathbf{R}$.

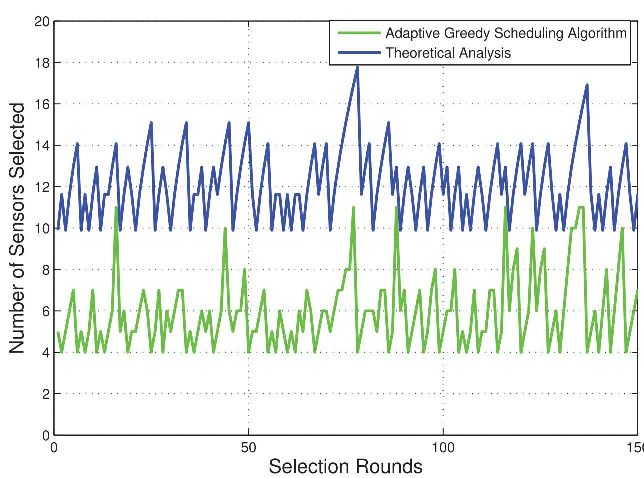


Fig. 5. Greedy results vs. theoretical results.

The time complexity of the greedy heuristic in the Cover Selection phase is $O(nm)$. From previous analysis, the time complexity of heuristics in the CPCP Enhancement phase is $O(n\mathbf{R}) = O(nm\frac{\ln(1-\beta^{\frac{1}{m}})}{\ln(1-w_{min})})$. Thus, the complexity of the proposed algorithm is $O(\tau k(nm + nm\frac{\ln(1-\beta^{\frac{1}{m}})}{\ln(1-w_{min})})) = O(\frac{\tau knm\ln(1-\beta^{\frac{1}{m}})}{\ln(1-w_{min})})$.

6. PERFORMANCE EVALUATION

6.1. Simulation Methodology and Setting

In this section, we evaluate our adaptive greedy scheduling algorithm. Our objective in conducting the evaluation study is threefold: (1) Testing the effectiveness of our scheme in providing required CPCP and prolonging network lifetime. (2) Evaluating the performance of our scheme under different system parameters, such as the ratio of regular sensors to robust sensors, different number of critical points, required CPCP, and preloaded energy in each sensor. (3) Studying the sensitivity of the algorithm to the weather forecast, including its accuracy, forecast period, and updating interval.

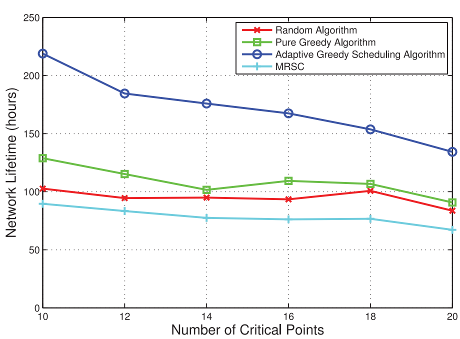
We use *network lifetime* as the primary metric to evaluate our algorithm. When there are not enough alive sensors to provide the required CPCP, we view the network as dead and we record the network lifetime. In addition, we define another metric, namely, CPCP ratio. To calculate CPCP ratio, we first calculate the CPCP provided by the selected sensors at the beginning of each time slot and divide it by the required CPCP. The ratio represents how redundant our selection is in this time slot. This ratio for each time slot will be averaged and we obtain the CPCP ratio. We define this metric to evaluate the effectiveness of our sensor selection. Ideally, we select a minimum number of sensors to provide the required CPCP to reduce energy consumption and the achieved CPCP is no more than and no less than the required one. However, considering the NP-complete nature of the problem, the objective is not achievable and we aim to reduce the ratio. The lower the CPCP ratio, the longer the network lifetime. Moreover, the metric of event detection percentage is adopted to evaluate our algorithm. The event detection percentage is defined as the number of total events detected by the sensors to the number of total events that happened in the network.

We compare our algorithm with the MRSC algorithm [He et al. 2010], a *random algorithm*, and a *pure greedy algorithm*. The MRSC algorithm is the closest work to ours. It is an energy efficient scheduling algorithm targeting on covering all critical points as well as maximizing network lifetime. It also incorporates sensors' failure model in the scheduling. Considering the weather information is not leveraged in He et al. [2010], we modify the MRSC algorithm to utilize the information for a fair comparison. We use a random algorithm and a pure greedy algorithm as a benchmark to evaluate the performance of our algorithm. In rainy weather, the two algorithms only schedule robust sensors, while in sunny weather, both regular and robust sensors can be scheduled. To reach the required CPCP, the random algorithm randomly selects sensors, while the pure greedy algorithm always chooses sensors that cover most critical points. Both of the algorithms are aware of weather forecast. The same parameters are used in all four algorithms.

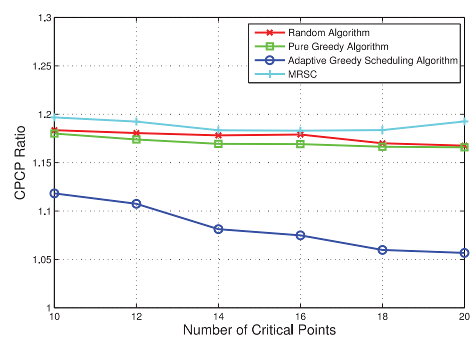
By default, there are 10 critical points in a $100m \times 100m$ square field in the simulation, unless we study the impact of the number of critical points. 200 sensor nodes are randomly deployed in the field. The ratio of the number of robust sensors to the number of regular sensors is 1 : 3 considering a robust sensor is much more expensive than a regular sensor. In each critical point, events happen randomly per time slot with the probability of 0.2. Unless we study the impact of weather forecast, we assume a weather report can be obtained every six hours. The report provides the precipitation chance for the next six hours. We set the forecast accuracy for the immediate next hour 90%. The accuracy decreases by 4% for each hour after that. Specifically, a weather forecast contains the precipitation chance for six hours. Its accuracies for the six hours are 90%, 86%, 82%, 78%, 74%, and 70%, respectively. All the evaluation results are the average of 100 experiments, each of which is on a different sensor distribution. The setting of the simulation is listed in Table I.

Table I. Simulation Configuration

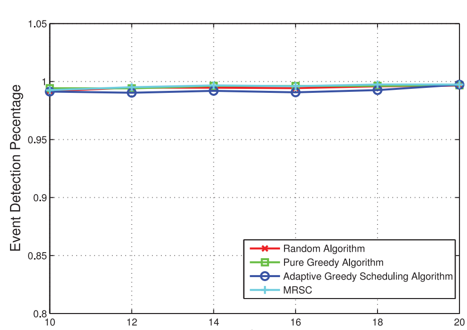
Parameter	Value
Sensing range (m)	20
Communication range (m)	40
Total number of sensors	200
Number of candidate sets selected in Cover Selection phase	5
Preloaded energy on each sensor (mAH)	1000
Energy consumption rate (on-duty) (mA)	8
Energy consumption rate (off-duty) (mA)	0.015
Number of critical points	10
Length of weather forecast	6
Frequency of weather forecast (*per hour*)	6
Accuracy of weather forecast	$70\% \sim 90\%$
Percentage of sunny periods and rainy periods in actual weather	80%, 20%
Failure rate (λ) of a robust sensor	1%
Failure rate (λ) of a regular sensor in rainy and non-rainy time	50%, 1%



(a) Network lifetime



(b) CPCP ratio

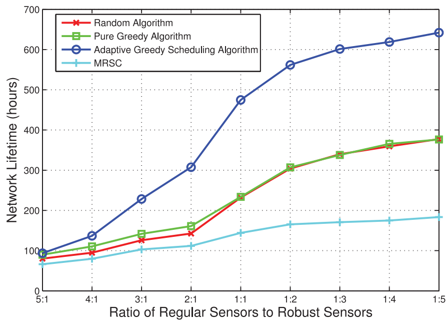


(c) Event detection percentage

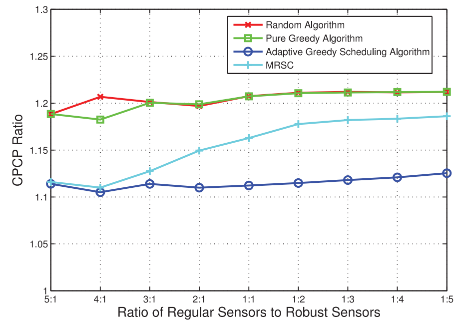
Fig. 6. Impact of number of critical points.

6.2. Impact of Number of Critical Points

In this evaluation, we increase the number of critical points from 10 to 20. Figure 6(a) shows the network lifetime under different number of critical points. From the figure, we can see that when the number of critical points increases, the network lifetime under all four algorithms decreases. The reason is obvious: a higher number of critical



(a) Network lifetime



(b) CPCP ratio

Fig. 7. Impact of different ratio.

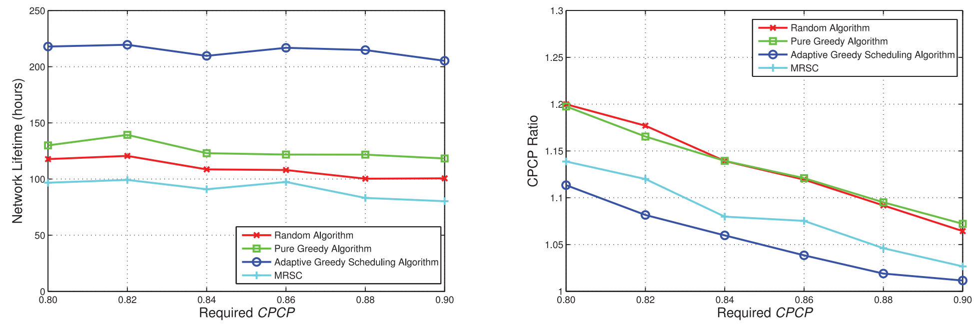
points demands more sensors to be on duty and thus the duration of being on duty for sensors becomes longer and the network lifetime consequently becomes shorter. Among the four algorithms, our algorithm has the longest network lifetime. For example, when the number of critical points is 20, the network lifetime is 135 hours. It is 101.5% longer than that under MSRC, which is 67 hours, 50.0% longer than that under the pure greedy algorithm, which is 90 hours, and 62.7% longer than that under the random algorithm, which is 83 hours.

Figure 6(b) shows that our algorithm achieves a lowest CPCP ratio among all algorithms. When the number of critical points is 20, the CPCP ratio under our algorithm is 1.05. It is less than that under the MSRC algorithm, which is 1.20. It is also less than that under the random algorithm and pure greedy algorithm, both of which are 1.17. As the number of critical points increases, the CPCP ratio decreases in our algorithms but remains almost the same in the other three algorithms. The reason is that our algorithm selects sensors based on the network topology. Thus, our algorithm performs more efficiently than the other three algorithms.

We further evaluate the event detection percentage under different number of critical points. Figure 6(c) shows that all four algorithms achieve a high event detection percentage, which is about 99%. Since the detection percentage always keeps a very high value under different parameters, in the following evaluations, we omit the evaluation of event detection percentage.

6.3. Impact of the Ratio of Regular Sensors to Robust Sensors

We vary the ratio of regular sensors to robust sensors from 5 : 1 to 1 : 5. Figure 7(a) shows the network lifetime under different ratios. From the figure, we can see that when the ratio is high, the performance of the four algorithms is close. This is because if the number of robust sensors is too small, when there is rain, there are not many options to schedule robust sensors. When the percentage of robust sensors increases, we have more choices in scheduling sensors in both sunny and rainy weather, and our algorithm can perform much better in prolonging the network lifetime. When the ratio is 1 : 5, the lifetime under our algorithm is the best, which is 640 hours. It is about 256% longer than the lifetime of the MRSC algorithm, which is 180 hours. It is also about 73% longer than the lifetime of the random algorithm and pure greedy algorithm, both of which are 371 hours. Since MRSC only schedules sensors based on their failure probability instead of choosing robust and regular sensors according to weather forecast, regular sensors fail more often in MRSC than in the other three schemes, and thus its performance is the worst.



(a) Network lifetime (b) CPCP ratio

Fig. 8. Survivability Threshold

Figure 7(b) shows the CPCP ratio. By default, the required CPCP is 80%. We can see that our CPCP ratio is closer to 1 than the other three algorithms, which means sensors are scheduled more efficiently in our algorithm than in the other algorithms. When the ratio is 1 : 5, the CPCP ratio under our algorithm is about 1.13. It is less than the CPCP ratio under the MSRC algorithm, which is 1.18. It is also less than the CPCP ratio under the pure greedy algorithm and random algorithm, both of which are 1.21. From results in Figure 7, it is clear that our proposed algorithm performs more efficiently than other algorithms in both network lifetime and CPCP ratio.

6.4. Impact of Required CPCP

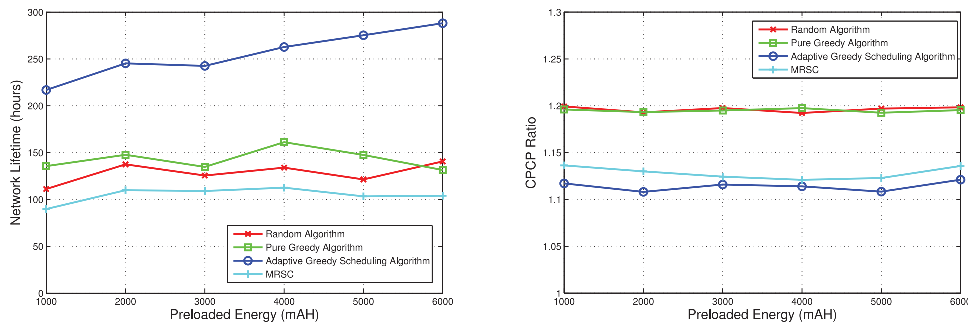
We vary the required CPCP from 0.8 to 0.9. Figure 8(a) shows the network lifetime under different CPCP. From the figure, we can see that when the required CPCP increases, the network lifetime under all four algorithms decreases. The reason is obvious: a higher required CPCP demands more sensors to be on duty and thus the duration of being on duty for sensors is longer and the network lifetime consequently is shorter. Among the four algorithms, the lifetime under our algorithm slightly decreases. When CPCP is 80%, the network lifetime is 220 hours; when the CPCP is 90%, the network lifetime is only 7% shorter. Moreover, the lifetime under our algorithm is twice as long as MSRC, 70% longer than the pure greedy algorithm, and 85% longer than random algorithm, when CPCP is 80%.

Figure 8(b) shows that as the required CPCP increases, the CPCP ratio decreases in all four algorithms. The reason is that as the CPCP increases, more sensors are needed in the scheduling. Then there are more selection options. Thus, the algorithm performs more efficiently. It can also be observed that our algorithm still performs the best. When required CPCP is 0.9, the CPCP ratio under our algorithm is 1.01. It is less than the CPCP ratio under the MSRC algorithm, which is 1.03. It is also less than the CPCP ratio under the random algorithm and pure greedy algorithm, both of which are 1.07.

6.5. Impact of Preloaded Energy

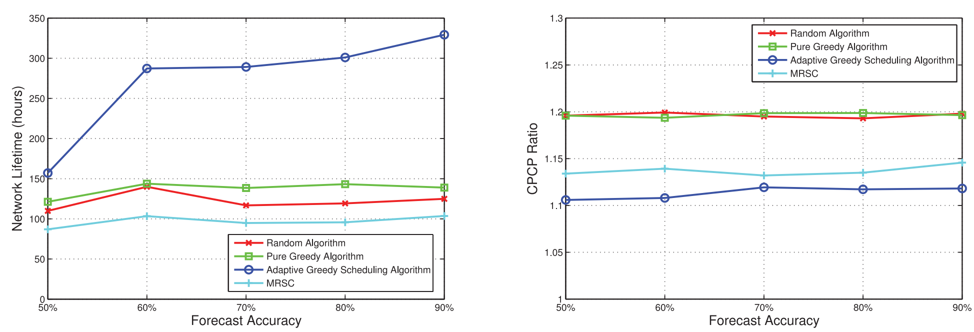
We vary the preloaded energy in each sensor from 1,000mAH to 6,000mAH. Figure 9 shows the network lifetime and CPCP ratio under different preloaded energy.

From Figure 9(a), we can see that the lifetime increases from 220 hours to 290 hours in our proposed algorithm. The network lifetime does not increase much for the other three algorithms. That is because, aside from drain of energy, a regular sensor can die because of rain. Since the weather report is inaccurate, regular sensors may be set on duty assuming no rain though it actually rains. The rain will destroy all the



(a) Network lifetime (b) CPCP ratio

Fig. 9. Energy Storage



(a) Network lifetime (b) CPCP ratio

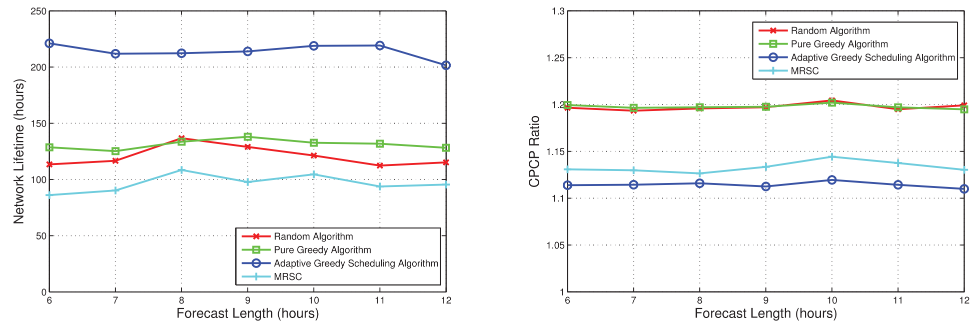
Fig. 10. Forecast accuracy.

regular sensors still having energy left. When a sensor has a high energy level, the weather forecast accuracy, instead of the energy consumption, starts to dominate the network lifetime. The network lifetime provided by our algorithm is still longer than the other three algorithms. When the preload energy is $6,000$mAH, the lifetime in our algorithm is 290 hours. It is about 2.8 times the lifetime in the MSRC algorithm, which is 105 hours. It is also about 2.1 times the lifetime in the random and pure greedy algorithms, both of which are 140 hours.

From Figure 9(b), it can be observed that preloaded energy does not affect the CPCP ratio, since energy is not considered in the sensors' selection.

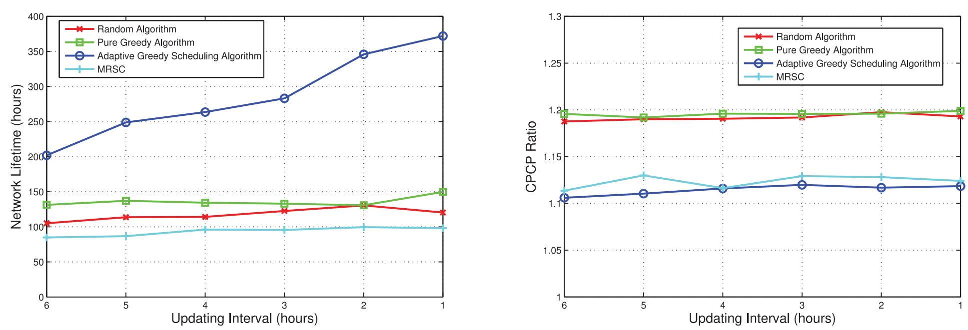
6.6. Impact of Weather Forecast

To show the impact of weather forecast, we evaluate both network lifetime and CPCP ratio under different accuracies, forecast periods, and updating frequencies of the weather forecast. We vary the accuracy of the weather forecast from 0.5 to 0.9 and plot the corresponding results in Figure 10. From Figure 10(a), we can see that when the accuracy increases, the network lifetime provided by our algorithm significantly increases, especially from 50% to 60%, which is from 158 hours to 288 hours, while the network lifetime provided by the other three algorithms barely changes. This proves our algorithm can fully leverage the weather forecast information and is able to correspondingly minimize and balance the network energy consumption. Our algorithm has the longest network lifetime in this simulation. When the accuracy is 90%, the lifetime under our algorithm is 330 hours. It is longer than the lifetime under the



(a) Network lifetime (b) CPCP ratio

Fig. 11. Forecast period.



(a) Network lifetime (b) CPCP ratio

Fig. 12. Updating period.

MSRC algorithm, which is 103 hours. It is also longer than the lifetime under the pure greedy algorithm, which is 139 hours, and longer than that under the random algorithm, which is 125 hours. Moreover, from Figure 10(b) it can be observed that the CPCP ratio does not much rely on an accurate weather forecast.

We vary the length of the weather report from 6 hours to 12 hours. When the length of the report is 12 hours, the report accuracy is 90%, 86%, 82%, 78%, 74%, 70%, 66%, 62%, 58%, 54%, 52%, and 50%, respectively. Figure 11(a) shows the network lifetime under different lengths of weather report. From the figure, we can see that increasing report length does not increase the network lifetime. That is because, the longer the report length, the less accuracy it has, which can cause more regular sensors to be scheduled in rainy time and thus failed. When the length of the weather forecast is 12 hours, our algorithm can achieve the lifetime for 201 hours, which is longer than that under the MSRC algorithm, the lifetime of which is 95 hours. It is also longer than the lifetime under the pure greedy algorithm, which is 128 hours, and longer than that under random algorithm, which is 115 hours. From Figure 11(b), it can be observed the CPCP ratio in our algorithm is still the one closest to the required CPCP. Besides, for all four algorithms, their CPCP ratios are not affected by longer weather forecast.

We vary the updating interval of the weather forecast from 6 hours to 1 hour. Figure 12(a) shows the network lifetime under different updating intervals. From this figure we can see that the lifetime under our algorithm increases when the weather is updated more frequently. This is because a more frequently updated weather forecast is more accurate. The figure shows our algorithm can benefit from the accurate weather forecast and perform better in scheduling the activities of sensors. Especially when the updating interval is 1 hour, the lifetime achieved by our algorithm is 372 hours, which is more than that by the MSRC algorithm, the lifetime of which is 100 hours. It is also longer than the lifetime under the random algorithm, which is 120 hours, and that under the pure greedy algorithm, which is 151 hours. Since the other three algorithms only use the weather information in the coming unit, their performance does not change much. Moreover, Figure 12(b) shows that the frequent weather forecast updating does not much affect the CPCP ratio in all four algorithms.

From all of the preceding results, we can conclude that our proposed algorithm can perform much more effectively and efficiently with an accurate and frequently updated weather forecast.

7. RELATED WORK

Duty-cycle scheduling in wireless sensor networks has been intensively studied in the literature. Depending on the network architecture, existing works can be classified into two categories: scheduling in *homogeneous* networks and scheduling in *heterogeneous* networks. In homogeneous networks, scheduling algorithms are designed to maximize the network lifetime given a set of critical locations to be covered [Cardei et al. 2005]. Under the assumption that a sensor can only cover a target at any time, the optimal schedule for the longest network lifetime can be derived by using matrix decomposition [Liu et al. 2005]. Scheduling mechanisms are also proposed in Tian and Georganas [2002], Wang et al. [2003], Zhang and Hou [2005], and Zou and Chakrabarty [2005] to trade the energy consumption for area coverage. Kasbekar et al. propose a scheduling algorithm that guarantees to cover the entire area of interest with a lower-bounded network lifetime [Kasbekar et al. 2009]. A scheduling proposed in Kumar et al. [2012] is to increase the lifetime of the network by saving the energy used in transmitting the redundant data from the nearby nodes. Paul et al. present a cluster-based scheduling considering mobility of nodes in Paul and Sao [2011]. Tian et al. propose a dynamic intruder detection scheme using mobile and static sensors to detect smart intruders [Tian et al. 2014a]. Scheduling for barrier coverage is studied in Kumar et al. [2010] and Tian et al. [2014c].

Different scheduling algorithms are designed in heterogeneous networks with regard to communication/sensing range [Gao et al. 2011] and energy storage [Kumara et al. 2009]. Lai et al. study heterogeneous asynchronous wake-up scheduling schemes for maintaining network connectivity while minimizing energy consumption [Lai et al. 2010]. Given a coverage requirement, Gupta et al. propose a scheduling protocol to identify redundant sensors to put into sleep [Gupta et al. 2013]. Ammari et al. propose scheduling protocols to achieve k-coverage in heterogeneous sensor networks [Ammari and Das 2011]. Shiha et al. consider each sensor equips multiple sensing units and design a scheduling protocol in wireless heterogeneous sensor networks [Shih et al. 2009]. Considering heterogeneity of real-time traffic over wireless channels, Hou et al. develop a general approach for designing scheduling policies [Hou and Kumar 2010]. Lee et al. propose a distributed scheduling for opportunistic forwarding under heterogeneous wake-up rate in each sensor to reduce end-to-end delay [Lee and Eun 2010].

In addition to scheduling, many other problems, such as coverage [Bartolini et al. 2012; Du and Lin 2005; Wang et al. 2011; Wu and Wang 2012], intrusion detection [Lazos et al. 2007; Wang et al. 2008], and data aggregation [Cardei et al. 2008; Chitnis et al. 2009], have been studied in heterogeneous networks. Machado et al. investigate the resource planning and study the general architecture of the heterogeneous networks with sensors having different survivability [Machado et al. 2009]. The network coverage is improved with mobility assistance in Wang et al. [2011]. The impact of sensing range and deployment topology on network coverage is studied in Du and Lin

[2005] and Wu and Wang [2012], respectively. An algorithm to reduce sensor coverage redundancy through joint sensor activation and sensing radius adaptation is proposed in Bartolini et al. [2012].

However, none of the existing works focuses on designing a scheduling algorithm in heterogeneous networks with sensors having different environmental survivability. While most of the existing scheduling works only consider the energy, none of them considers environmental attributes, such as weather, which is an important factor that determines the network lifetime. Without such consideration, the schemes cannot schedule sensors to cope with different weather and thus cannot be directly applied to solve the problem in this article.

8. CONCLUSION

In this article, we for the first time study the scheduling problem in heterogeneous sensor networks regarding environmental survivability. We theoretically prove the problem is NP-complete and propose a novel adaptive greedy algorithm to solve the problem. The algorithm schedules sensors' activities based on their survivability, remaining energy and coverage, and the weather forecast information.

REFERENCES

H. M. Ammari and S. K. Das. 2011. Scheduling protocols for homogeneous and heterogeneous-covered wireless sensor networks. *Pervasive and Mobile Computing* 7, 1 (2011), 79–97.

S. Aschenbrenner. 2006. Failure modes, effects and diagnostic analysis. *Channels*.

N. Bartolini, T. Calamoneri, T. La Porta, C. Petrioli, and S. Silvestri. 2012. Sensor activation and radius adaptation (SARA) in heterogeneous sensor networks. *ACM Transactions on Sensor Networks* 8, 3 (Aug. 2012), 24:1–24:34.

M. Cardei, M. T. Thai, Y. Li, and W. Wu. 2005. Energy-efficient target coverage in wireless sensor networks. In *IEEE INFOCOM*, 3. 1976–1984.

M. Cardei, S. Yang, and J. Wu. 2008. Algorithms for fault-tolerant topology in heterogeneous wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 19 (2008), 545–558.

A. Chen, Z. Li, T. H. Lai, and C. Liu. 2011a. One-way barrier coverage with wireless sensors. In *IEEE INFOCOM*. 626–630.

T. Chen, Z. Yang, Y. Liu, D. Guo, and X. Luo. 2011b. Localization in non-localizable sensor and ad-hoc networks: A localizability-aided approach. In *IEEE INFOCOM*. 276–280.

L. Chitnis, A. Dobra, and S. Ranka. 2009. Fault tolerant aggregation in heterogeneous sensor networks. *Journal of Parallel and Distributed Computing* 69 (2009), 210–219.

X. Du and F. Lin. 2005. Maintaining differentiated coverage in heterogeneous sensor networks. *EURASIP Journal on Wireless Communications and Networking*. 4 (2005), 565–572.

A. Enyedi. 2010. What are PoPs? http://www.srh.noaa.gov/jax/?n=probability_of_precipitation.

L. Gao, X. Wang, G. Sun, and Y. Xu. 2011. A game approach for cell selection and resource allocation in heterogeneous wireless networks. In *IEEE SECON*. 530–538.

W. M. Goble. 2002. Getting Failure Rate Data. www.exida.com.

H. P. Gupta, S. V. Rao, and T. Venkatesh. 2013. Sleep scheduling for partial coverage in heterogeneous wireless sensor networks. In *Communication Systems and Networks*. 1–10.

J. He, N. Xiong, Y. Xiao, and Y. Pan. 2010. A reliable energy efficient algorithm for target coverage in wireless sensor networks. In *IEEE International Conference on Distributed Computing Systems Workshops*. 180–188.

G. Hoblos, M. Staroswiecki, and A. Aitouche. 2000. Optimal design of fault tolerant sensor networks. In *IEEE International Conference on Control Applications*. 467–472.

I. H. Hou and P. R. Kumar. 2010. Scheduling heterogeneous real-time traffic over fading wireless channels. In *IEEE INFOCOM*. 1–9.

X. Hu, Y. H. Hu, and B. Xu. 2014. Energy-balanced scheduling for target tracking in wireless sensor networks. *ACM Transactions on Sensor Networks (TOSN)* 11, 1 (2014), 21:1–21:29.

M. Huang, S. Chen, and Y. Wang. 2010. Minimum cost localization problem in wireless sensor networks. In *IEEE SECON*. 1–9.

W. Kang, J. A. Stankovic, and S. H. Son. 2008. On using weather information for efficient remote data collection in WSN. *Workshop on Embedded Networked Sensors*.

G. S. Kasbekar, Y. Bejerano, and S. Sarkar. 2009. Lifetime and coverage guarantees through distributed coordinate-free sensor activation. In *ACM Mobicom*.

R. Kumar, G. Dua, and S. Chaudhary. 2012. Group based energy efficient scheduling in wireless sensor networks. *International Journal of Advanced Research in Computer Science and Software Engineering* 2 (2012), 7.

S. Kumar, T. H. Lai, M. E. Posner, and P. Sinha. 2010. Maximizing the lifetime of a barrier of wireless sensors. *IEEE Transactions on Mobile Computing* 9, 8 (2010), 1161–1172.

D. Kumara, T. C. Aserib, and R. B. Patelc. 2009. EEHC: Energy efficient heterogeneous clustered scheme for wireless sensor networks. *Computer Communications* 32 (2009), 662–667.

S. Lai, B. Ravindran, and H. Cho. 2010. Heterogenous quorum-based wake-up scheduling in wireless sensor networks. *IEEE Transactions on Computers* 59, 11 (2010), 1562–1575.

L. Lazos, R. Poovendran, and J. A. Ritcey. 2007. Probabilistic detection of mobile targets in heterogeneous sensor networks. *Proceedings of the 6th International Conference on Information Processing in Sensor Networks*, 519–528.

C. H. Lee and D. Y. Eun. 2010. A distributed wake-up scheduling for opportunistic forwarding in wireless sensor networks. In *IEEE GLOBECOM*. 1–5.

H. Liu, P. Wan, C. W. Yi, X. Jia, S. Makki, and N. Pissinou. 2005. Maximal lifetime scheduling in sensor surveillance networks. In *IEEE INFOCOM*, Vol. 4. 2482–2491.

W. Liu, D. Wang, H. Jiang, W. Liu, and C. Wang. 2012. Approximate convex decomposition based localization in wireless sensor networks. In *IEEE INFOCOM*. 1853–1861.

A. C. Lorenc. 1986. Analysis methods for numerical weather prediction. *Quarterly Journal of the Royal Meteorological Society*.

R. Machado, W. Zhang, and G. Wang. 2009. Network planning for heterogeneous wireless sensor networks in environmental survivability. *IEEE ICTAI* 0, 814–821.

K. Martinez, J. K. Hart, and R. Ong. 2004. Environmental sensor networks. *Computer*.

S. Paul and N. K. Sao. 2011. An energy efficient hybrid node scheduling scheme in cluster based wireless sensor networks. *Proceedings of the World Congress on Engineering*.

R. Peck and J. L. Devore. 2013. *Statistics: The Exploration and Analysis of Data*. Cengage Learning, 6th edition.

L. F. Richardson. 2007. *Weather Prediction by Numerical Process*. Cambridge University Press.

L. B. Ruiz, I. G. Siqueira, L. B. Oliveira, H. C. Wong, J. M. S. Nogueira, and A. A. F. Loureiro. 2004. Fault management in event-driven wireless sensor networks. In *Proceedings of the 7th ACM International Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems*.

L. Selavo, A. Wood, Q. Cao, T. Sookoor, H. Liu, A. Srinivasan, Y. Wu, W. Kang, J. Stankovic, D. Young, and J. Porter. 2007. LUSTER: Wireless sensor network for environmental research. In *Proceedings of the 5th International Conference on Embedded Networked Sensor Systems*. 103–116.

K. P. Shih, H. C. Chen, C. M. Chou, and B. J. Liu. 2009. On target coverage in wireless heterogeneous sensor networks with multiple sensing units. *Journal of Network and Computer Applications* 32 (2009), 866–877.

D. Tian and N. Georganas. 2002. A coverage-preserving node scheduling scheme for large wireless sensor networks. In *ACM International Workshop on Wireless Sensor Networks and Applications*.

J. Tian, G. Wang, T. Yan, and W. Zhang. 2014a. Detect smart intruders in sensor networks by creating network dynamics. *Computer Networks* 62 (2014), 31–42.

J. Tian, T. Yan, and G. Wang. 2014b. A network coding based energy efficient data backup in survivability-heterogeneous sensor networks. *IEEE Transactions on Mobile Computing*, available online.

J. Tian, W. Zhang, G. Wang, and X. Gao. 2014c. 2D k-barrier duty-cycle scheduling for intruder detection in wireless sensor networks. *Computer Communications* 43 (2014), 182–196.

X. Wang, X. Wang, and J. Zhao. 2011. Impact of mobility and heterogeneity on coverage and energy consumption in wireless sensor networks. In *ICDCS*. 477–487.

X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill. 2003. Integrated coverage and connectivity configuration in wireless sensor networks. In *ACM SenSys*.

Y. Wang, X. Wang, B. Xie, D. Wang, and D. P. Agrawal. 2008. Intrusion detection in homogeneous and heterogeneous wireless sensor networks. *IEEE Transactions on Mobile Computing* 7 (2008), 698–711.

D. West. 2001. *Introduction to Graph Theory* (2nd ed.). Prentice Hall.

Y. Wu and X. Wang. 2012. Achieving full view coverage with randomly-deployed heterogeneous camera sensors. In *ICDCS*. 556–565.

H. Zhang and J. C. Hou. 2005. Maintaining sensing coverage and connectivity in large sensor networks. In *Wireless Ad Hoc and Sensor Networks*.

Y. Zou and K. Chakrabarty. 2005. A distributed coverage and connectivity centric technique for selecting active nodes in wireless sensor networks. In *IEEE Transactions on Computers* 54, 8 (2005), 978–991.

Received March 2014; revised December 2014; accepted April 2015